

Discussion 1: Algorithm Analysis

April 8

Caijie Zhang

<http://www.cs.ucsb.edu/~caijie>

1. Order the following functions by growth rate: N , $N^{0.5}$, $N^{1.5}$, N^2 , $N \log N$, $N \log \log N$, $N \log^2 N$, $N \log(N^2)$, $2/N$, 2^N , $2^{N/2}$, 37 , $N^2 \log N$, N^3 . Indicate which functions grow at the same rate.

Solution: $2/N$, 37 , $N^{0.5}$, N , $N \log \log N$, $(N \log N, N \log(N^2))$, $N \log^2 N$, $N^{1.5}$, N^2 , $N^2 \log N$, N^3 , $2^{N/2}$, 2^N

2. Suppose $T_1(N) = O(f(N))$ and $T_2(N) = O(f(N))$. Which of the following are true?
- $T_1(N) + T_2(N) = O(f(N))$
 - $T_1(N) - T_2(N) = o(f(N))$
 - $T_1(N) / T_2(N) = O(1)$
 - $T_1(N) = O(T_2(N))$

Solution: a

a. proof:

As $T_1(N) = O(f(N))$, there are positive const n_1 and c_1 such that $T_1(N) \leq c_1 f(N)$ when $n \geq n_1$.

As $T_2(N) = O(f(N))$, there are positive const n_2 and c_2 such that $T_2(N) \leq c_2 f(N)$ when $n \geq n_2$.

Let $n_3 = \max(n_1, n_2)$ and $c_3 = c_1 + c_2$, then we have

$$T_1(N) + T_2(N) \leq (c_1 + c_2) f(N) = c_3 f(N) \text{ when } n \geq n_3.$$

b. counter example:

$$T_1(N) = 2n, T_2(N) = n, f(N) = n$$

$$T_1(N) - T_2(N) = n$$

c. counter example

$$T_1(N) = n^2, T_2(N) = n, f(N) = n^2$$

$$T_1(N) / T_2(N) = n$$

d. counter example

$$T_1(N) = n^2, T_2(N) = n, f(N) = n^2$$

3. Consider the following Interval Coloring Problem.

INPUT: A set $S = \{(x_i, y_i) \mid 1 \leq i \leq n\}$ of intervals over the real line. Think of interval (x_i, y_i) as being a request for a room for a class that meets from time x_i to time y_i .

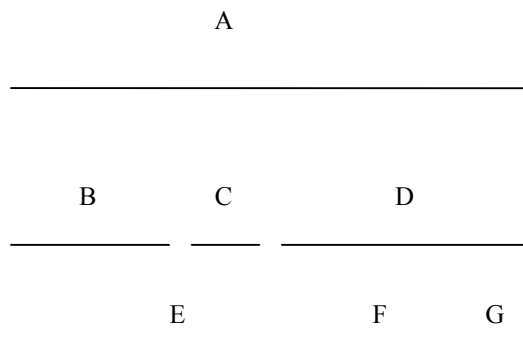
OUTPUT: Find an assignment of classes to rooms that uses the fewest number of rooms.

Note that every room request must be honored and that no two classes can use a room at the same time.

- (a) Consider the following iterative algorithm. Assign as many classes as possible to the first room (we can do this using the greedy algorithm discussed in class, and in the class notes), then assign as many classes as possible to the second room, then assign as many classes as possible to the third room, etc. Does this algorithm solve the Interval Coloring Problem? Justify your answer.
- (b) Consider the following algorithm. Process the classes in increasing order of start times. Assume that you are processing class C . If there is a room R such that R has been assigned to an earlier class, and C can be assigned to R without overlapping previously assigned classes, then assign C to R . Otherwise, put C in a new room. Does this algorithm solve the Interval Coloring Problem? Justify your answer.

Solution:

- (a) This algorithm does not solve the interval-coloring problem. Consider the following intervals:



The optimal solution is to put A in one room, $\{B,C,D\}$ in another, and $\{E,F,G\}$ in another, for a total of 3 rooms. However, maximizing the number of classes in the first room results in having $\{B,C,F,G\}$ in one room, and classes A , D and G each in their own rooms, for a total of 4.

(b) This algorithm does solve the interval-coloring problem.

Proof:

Definition: We define the depth of a set of intervals to be the maximum number that pass over any single point on the time-line.

Claim 1: In any instance of Interval Partitioning, the number of resources needed is at least the depth of the set of intervals

Let d be the depth of the set of intervals; we show how to assign a label to each interval, where the labels come from the set of numbers $\{1, 2, \dots, d\}$, and the assignment has the property that overlapping intervals are labeled with different numbers.

Claim 2: If we use the greedy algorithm (b), every interval will be assigned a label and no two overlapping intervals will receive the same label.

Since the optimal solution must use at least d labels to color all the classes and algorithm (b) can solve this problem using d labels, we conclude that algorithm (b) always using the minimum possible number of labels.